Introducción al lenguaje JAVA

1. Instalación

Instalar el **compilador de Java** y la **máquina virtual de Jav**a. Descargar desde:
http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

Opcionalmente se puede instalar el IDE **Eclipse**.
http://www.eclipse.org/downloads/

2. Consideraciones Generales

2.1 Comentarios

En java los comentarios se ponen entre /* */

2.2 Nombres de clases, métodos y variables

En las clases cada palabra debe comenzar con Mayuscula.
Ejemplo: NombreDeMiClase

En los métodos cada palabra debe comenzar con Mayuscula excepto la primera.
Ejemplo: nombreDeMiMetodo

Las variables van siempre en mininuscula.
Ejemplo: variable1

Cada bloque de código va entre llaves ({}).

Todas las sentencias terminan con punto y coma (;)

3. Operadores

3.1 Operadores relacionales

> (mayor)
< (menor)
>= (mayor o igual)
<= (menor o igual)
== (igual)
!= (distinto)

3.2. Operadores Matemáticos

+ (más)
- (menos)
* (producto)
/ (división)

% (resto de una división)

3.3 Operadores Lógicos

&& (y)

|| (o)

4. Condicional (IF / ELSE)

```
if (num1>num2) {
        System.out.print(num1);
else if (num1==num2) {
        System.out.print(“Iguales”);
} else {
        System.out.print(num2);
}
```

5. Estructuras repetitivas

5.1 Estructura repetitiva while

```
public class EstructuraRepetitivaWhile {
    public static void main(String[] ar) {
        int x;
        x=1;
        while (x<=100) {
            System.out.print(x);
            System.out.print(" - ");
            x = x + 1;
        }
    }
}
```

5.2. Estructura repetitiva for

```
public class EstructuraRepetitivaFor {
    public static void main(String[] ar) {
        int f;
        for(f=1;f<=100;f++) {
            System.out.print(f);
            System.out.print("-");
        }
    }
}
```

5.3 Estructura repetitiva do while

```
do {
} while (CONDICIÓN);
```

6. Tipos de datos

```
byte dato = 25;
short dato = 2500;
int dato = 250000;
long dato = 2500000000L;
float dato = 250.56f;
double dato = 2500000.5467
boolean dato = true;
char dato = 'A'; //Siempre comillas simples
String dato = "El primer programa";
```

7. Declaración de una clase y definición de objetos.

Una clase es un molde del que luego se pueden crear múltiples objetos, con similares características. Una clase es una plantilla (molde), que define atributos (variables) y métodos (funciones)

Estructura de una clase

```
class [nombre de la clase] {
  [atributos o variables de la clase]
  [métodos o funciones de la clase]
  [main]
}
```

Ejemplo:

Crear una clase que permita carga el nombre y la edad de una persona. Mostrar los datos cargados. Imprimir un mensaje si es mayor de edad.

```
import java.util.Scanner;
public class Persona {
   private Scanner teclado;
   private String nombre;
   private int edad;
   public void inicializar() {
      teclado=new Scanner(System.in);
      System.out.print("Ingrese nombre:");
      nombre=teclado.next();
      System.out.print("Ingrese edad:");
      edad=teclado.nextInt();
   }
   public void imprimir() {
      System.out.println("Nombre:"+nombre);
      System.out.println("Edad:"+edad);
   }

   public void esMayorEdad() {
      if (edad>=18) {
         System.out.print(nombre+" es mayor de edad.");
      } else {
         System.out.print(nombre+" no es mayor de edad.");
```

```
        }
    }

    public static void main(String[] ar) {
        Persona persona1;
        persona1=new Persona();
        persona1.inicializar();
        persona1.imprimir();
        persona1.esMayorEdad();
    }
}
```

- El nombre de la clase debe hacer referencia al concepto (en este caso la hemos llamado Persona):
- Los atributos los definimos dentro de la clase pero fuera de la main:
- A los atributos se tiene acceso desde cualquier función o método de la clase (salvo la main)
- Luego de definir los atributos de la clase debemos declarar los métodos o funciones de la clase. La sintaxis es parecida a la main (sin la cláusula static):

7.1 Declaración de métodos

7.1.1 Métodos con parametros

```
    public void calcular(int v) {
        for(int f=v;f<=v*10;f=f+v) {
            System.out.print(f+"-");
        }
    }
```

7.1.1 Métodos que retornan un dato

```
        public int calcularMenor(int v1,int v2,int v3) {
        ...
        }
```

7.2 Herencia

La herencia significa que se pueden crear nuevas clases partiendo de clases existentes, que tendrá todas los atributos y los métodos de su 'superclase' o 'clase padre' y además se le podrán añadir otros atributos y métodos propios.

```
public class Operacion {
    protected Scanner teclado;
    protected int valor1;
    protected int valor2;
    protected int resultado;
    public Operacion() {
        teclado=new Scanner(System.in);
    }

    public void cargar1() {
        System.out.print("Ingrese el primer valor:");
        valor1=teclado.nextInt();
    }

    public void cargar2() {
        System.out.print("Ingrese el segundo valor:");
```

```
        valor2=teclado.nextInt();
    }

    public void mostrarResultado() {
        System.out.println(resultado);
    }
}
public class Suma extends Operacion{
    void operar() {
        resultado=valor1+valor2;
    }
}
public class Resta extends Operacion {
    public void operar(){
        resultado=valor1-valor2;
    }
}

public class Prueba {
    public static void main(String[] ar) {
        Suma suma1=new Suma();
        suma1.cargar1();
        suma1.cargar2();
        suma1.operar();
        System.out.print("El resultado de la suma es:");
        suma1.mostrarResultado();
        Resta resta1=new Resta();
        resta1.cargar1();
        resta1.cargar2();
        resta1.operar();
        System.out.print("El resultado de la resta es:");
        resta1.mostrarResultado();
    }
}
```